Centre de recherche sur les transports (C.R.T.)
Centre for Research on Transportation

CRT-2006-26

A SCATTER SEARCH HEURISTIC FOR THE FIXED-CHARGE CAPACITATED NETWORK DESIGN PROBLEM

by

Teodor Gabriel Crainic
Michel Gendreau

November 2006

Université de Montréal
C.P. 6128, succursale Centre-ville, Montréal QC H3C 3J7 Canada
Téléphone: (514) 343-7575 / Télécopieur: (514) 343-7121

crt@crt.umontreal.ca
www.crt.umontreal.ca

HEC ÉCOLE POLYTECHNIQUE MONTRÉAL Université de Montréal

2007-00275

PUBLICATION
CRT-2006-26

A SCATTER SEARCH HEURISTIC FOR THE FIXED-CHARGE CAPACITATED NETWORK DESIGN PROBLEM

by

Teodor Gabriel Crainic[1]
Michel Gendreau[2]

The research was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

1. Département de management et technologie, École des sciences de la gestion, Université du Québec à Montréal, C.P. 8888, succursale Centre-ville, Montréal, Canada H3C 3P8 and Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT), Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Canada H3C 3J7 (theo@crt.umontreal.ca)
2. Département d'informatique et de recherche opérationnelle and Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT), Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Canada H3C 3J7 (michelg@crt.umontreal.ca)

Dépôt légal – Bibliothèque nationale du Québec
Bibliothèque nationale du Canada, 2006

Abstract

The fixed-charge Capacitated Multi-commodity Network Design (CMND) problem consists in finding the optimal configuration, i.e., the arcs to include in the final design, of a network on which the flows of several products (commodities) must be routed to satisfy given demands between origin-destination pairs. Each of the arcs that can possibly be included in the design is characterized by its capacity (the maximum amount of flow of all commodities it can support), a fixed cost to be incurred if the arc is selected, and a variable cost for each unit of flow that uses the arc. The objective of the problem is to minimize the total system cost (the sum of the fixed costs of selected arcs and routing costs), while respecting capacity limits.

In this paper, we report on an extensive investigation of different variants of a new metaheuristic, based on the Scatter Search concept originally proposed by Glover, for the CMND. Computational results on a set of small and medium size benchmark instances show that while scatter search is not yet able to match the results of the best existing metaheuristics for the problem, all variants are successful in finding better solutions on some instances.

Keywords: scatter search, network design, multi-commodity flows, fixed-charge problems.

Résumé

Le problème de synthèse de réseaux multiproduits à coûts fixes consiste à trouver une configuration optimale, c.-à-d. la liste des arcs à retenir, pour un réseau sur lequel les flux de plusieurs produits doivent être acheminés pour satisfaire des demandes connues entre des paires origine-destination. Chaque arc pouvant être inclus dans la configuration finale est caractérisé par sa capacité (le flux maximal de tous les produits qu'il peut acheminer), un coût fixe à encourir s'il est choisi, et un coût variable par unité de flux qui l'emprunte. L'objectif du problème est de minimiser le coût total du système (somme des coûts fixes des arcs choisis et des coûts variables d'acheminement), tout en respectant les contraintes de capacité sur les arcs.

Dans cet article, nous présentons les résultats d'une étude exhaustive de plusieurs variantes d'une nouvelle métaheuristique, basée sur le concept de « Scatter Search » proposé par Glover, pour ce problème de synthèse de réseau. Les résultats numériques obtenus sur jeu de problèmes tests de petite et moyenne taille indiquent que bien que la nouvelle métaheuristique ne soit pas tout à fait aussi performante que les meilleures disponibles, toutes ses variantes réussissent à trouver de meilleures solutions sur certaines instances.

Mots-clés : « scatter search », synthèse de réseau, flux multiproduits, problèmes à coûts fixes.

1 Introduction

The fixed-charge capacitated multicommodity network design problem (CMND) is a generic model that covers a wide range of network planning problems in the areas of transportation, logistics, telecommunication, and production management [2, 14, 15]. The problem consists in finding the optimal configuration, i.e., the arcs to include in the final design, of a network on which the flows of several products (or "commodities") must be routed to satisfy given demands between origin-destination pairs. Each of the arcs that can possibly be included in the design is characterized by its capacity (the maximum amount of flow of all commodities it can support), a fixed cost to be incurred if the arc is selected, and a variable cost for each unit of flow that uses the arc. The objective of the problem is to minimize the total system cost, computed as the sum of the fixed costs of selected arcs and routing costs, while respecting capacity limits.

The CMND problem is usually modeled as a 0–1 mixed integer programming problem and it has been shown to be NP-hard in the strong sense. Not surprisingly, even though significant efforts have been devoted to the development of exact methods for this problem (see, e.g., [3, 5, 12]), heuristics must be resorted to when dealing with large instances with several commodities.

Over the last few years, two main approaches based on metaheuristics have been proposed for the general CMND model. The first, which was developed by Crainic, Gendreau, and Farvolden [4], is a tabu search heuristic for the path-based formulation of the problem. It exploits the fixed-charge nature of the problem by exploring the space of the continuous path-flow variables using pivot-like moves in a column generation environment. This method produces impressive results compared to simple heuristics, but its efficiency remains limited by the fact that each move considers the impact of changing the flow of only one commodity (a pivot from one path to another), thus making it difficult to properly account for the multi-commodity nature of the problem. To overcome this limitation, Ghamlouche, Crainic, and Gendreau [6] proposed another tabu search heuristic, but for the arc-based formulation of the problem. The key element of this heuristic was a new neighbourhood structure for the CMND, the so-called "cycle-based neighbourhood", that allows changing the flow pattern of several commodities simultaneously, as well as opening and closing several arcs. In spite of the fact that the global search strategy used in this paper amounted to a fairly basic tabu search scheme, it was at the time the best approximate solution method for the CMND in terms of robust performance, solution quality, and computing efficiency. The authors extended their approach in a second paper [7] in which tabu search was combined with Path Relinking [9, 10, 11] to yield an enhanced and more effective search strategy. As far as we know, this is currently the most effective method for tackling large CMND instances. However, for some of the larger and more complex instances, the relative gaps observed between the lower bounds computed using Lagrangian relaxation [3, 5] and the best solutions obtained by the combined tabu search-path relinking hybrid heuristic can be as large as 20.9%, thus suggesting that there might exist significantly better solutions than these. It is therefore relevant to pursue examining other heuristic approaches for the CMND.

The purpose of this paper is to report on an extensive investigation of different variants of a new metaheuristic for the CMND. This new heuristic is based on the Scatter Search concept originally introduced by Glover [8, 9].

The remainder of the paper is organized as follows. In section 2, we recall the arc-based formulation of the CMND, as well as some of its basic properties that will be exploited in our scatter search heuristic. Following a brief outline of the scatter search methodology, section 3 details our scatter search implementation for the CMND. Computational results are reported and analyzed in section 4; we compare, in particular, the scatter search results to those obtained by the path relinking hybrid. Section 5 concludes the paper and suggests future research directions for the application of scatter search to network design problems.

2 Formulation and basic properties

Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ be a network with set of nodes $\mathcal{N}$ and set of directed arcs $\mathcal{A}$. Let $\mathcal{P}$ denote the set of commodities to move using this network and for each $p \in \mathcal{P}$, let d^p denote the required amount of flow of commodity p to be shipped from its origin $o(p)$ to its destination $s(p)$. The total flow on each arc $(i,j) \in \mathcal{A}$ is limited by the capacity u_{ij}. There are two costs involved in the network. The unit cost of moving commodity $p \in \mathcal{P}$ through the arc (i,j), denoted c_{ij}^p, and the fixed cost of including arc (i,j) in the design of the network, denoted f_{ij}. The problem consists in minimizing the sum of all costs while satisfying the demand of transportation.

The arc-based formulation of the CMND can then be written as

$$\min z(x,y) = \sum_{(i,j)\in\mathcal{A}} f_{ij}y_{ij} + \sum_{p\in\mathcal{P}}\sum_{(i,j)\subset\mathcal{A}} c_{ij}^p x_{ij}^p \tag{1}$$

$$\text{subject to} \sum_{j\in\mathcal{N}^+(i)} x_{ij}^p - \sum_{j\in\mathcal{N}^-(i)} x_{ji}^p = \begin{cases} d^p & if\ i = o(p) \\ -d^p & if\ i = s(p) \\ 0 & \text{otherwise.} \end{cases} \quad \forall i \in \mathcal{N}, \forall p \in \mathcal{P}, \tag{2}$$

$$\sum_{p\in\mathcal{P}} x_{ij}^p \leq u_{ij}y_{ij} \quad \forall (i,j) \in \mathcal{A}, \tag{3}$$

$$x_{ij}^p \geq 0 \quad \forall (i,j) \in \mathcal{A}, \forall p \in \mathcal{P}, \tag{4}$$

$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in \mathcal{A} \tag{5}$$

where y_{ij}, $(i,j) \in \mathcal{A}$, represent the design variables that equal 1 if arc (i,j) is selected in the final design (and 0 otherwise), x_{ij}^p stand for the flow distribution decision variables indicating the amount of flow of commodity $p \in \mathcal{P}$ on arc (i,j), and $\mathcal{N}^+(i)/\mathcal{N}^-(i)$ denotes the set of outward/inward neighbours of node i.

The objective function (1) accounts for the total system cost, the fixed cost of arcs included in a given design plus the cost of routing the demand of all commodities, and aims

to select the minimum cost design. Constraints (2) represent the network flow conservation relations, while constraints (3) state that for each arc, the total flow of all commodities cannot exceed its capacity if the arc is opened ($y_{ij} = 1$) and must be 0 if the arc is closed ($y_{ij} = 0$). Relations (4) and (5) are the usual non-negativity and integrality constraints for decision variables.

For a given design vector $\bar{y}$, the arc-based formulation of the CMND becomes a capacitated multicommodity minimum cost flow problem (CMCF)

$$\min\ z(x(\bar{y})) = \sum_{p \in \mathcal{P}} \sum_{(i,j) \in \mathcal{A}(\bar{y})} c_{ij}^p x_{ij}^p \tag{6}$$

subject to (2) plus

$$\begin{aligned} \sum_{p \in \mathcal{P}} x_{ij}^p &\leq u_{ij} \bar{y}_{ij} \quad \forall (i,j) \in \mathcal{A}(\bar{y}), \\ x_{ij}^p &\geq 0 \quad \forall (i,j) \in \mathcal{A}(\bar{y}), \forall p \in \mathcal{P}, \end{aligned}$$

where $\mathcal{A}(\bar{y})$ stands for the set of arcs corresponding to the design $\bar{y}$. A solution to the CMND may thus be viewed as an assignment $\bar{y}$ of 0 or 1 to each design variable, plus the optimal flow pattern of the corresponding multicommodity minimum cost flow problem $x^*(\bar{y})$. Similarly, the objective function value associated with a solution $(\bar{y}, x^*(\bar{y}))$ is the sum of the fixed cost of the open arcs in $\bar{y}$ and the objective function value of the CMCF associated with $\bar{y}$

$$z(\bar{y}, x^*(\bar{y})) = \sum_{(i,j) \in \mathcal{A}(\bar{y})} f_{ij} \bar{y}_{ij} + z(x^*(\bar{y})). \tag{7}$$

3 Scatter search

Scatter search is a population-based search heuristic that was originally introduced by Glover in 1977 [8]. The basic idea of the method is to create new (hopefully) *interesting* solutions to a problem by combining values from *elite* solutions previously obtained either by some other (meta-)heuristic or by the method itself. As in genetic algorithms, the key idea behind the method is that the best solutions to a problem must share some common attributes. Scatter search is particularly well-suited to problems with continuous decision variables, since elite solutions can then be combined through linear intra- or extrapolation. However, it can also be applied to combinatorial problems, but this requires some ingenuity in the definition of the procedures used to combine elite solutions, as we shall see in the following. The basic template for scatter search as proposed by Glover in 1998 [9] is as follows:

Scatter search template

1. Generate an initial population of *good* and *diverse* solutions.
2. Select a subset of the population to form the *Reference Set* (RS).
3. Extract N solutions from RS to create the *Candidate Set* (CS).
4. Create new solutions by combining the solutions from CS.
5. If necessary, repair these solutions to make them feasible.
6. Improve the new solutions.
7. Update the reference set and go back to step 3.

Interested readers will find more details on scatter search in the original papers by Glover [8, 9] or in the book by Laguna and Martí [13].

Our implementation of scatter search for the CMND focuses on the y_{ij} *design* variables; the continuous flow variables are computed by solving the associated CMCF subproblem. The initial population of solutions is obtained by applying the cycle-based tabu search of Ghamlouche, Crainic, and Gendreau [6] until it has performed a pre-set number of iterations without improvement (20, in the current implementation). The reference set is made up of S local optima extracted from the initial population. If more than S local optima are identified, we keep the S best ones; if there are less than S local optima, we complete RS with the best solutions encountered by tabu search that were not local optima. At each iteration, the candidate set is created by selecting the best solution in RS, along with the one that is farthest from it; if $N > 2$, CS is completed by randomly chosen solutions in RS.

We create a single new solution from the candidate set. To combine solutions $y^l, 1, ..., N$, of CS, we first compute for each arc (i, j) a *desirability factor*, $0 \leq m_{ij} \leq 1$, :

$$m_{ij} = \frac{\sum_l w_l y^l_{ij}}{\sum_l w_l}, \forall (i, j),$$

where w_l represents the *weight* of solution y^l. Three alternate variants were examined to define the weights:

- Voting (V): $w_l = 1, \forall l$;
- Cost (C): $w_l = 1/$(cost difference between solution l and the best solution), $\forall l$;
- Distance (H): $w_l = 1/$(Hamming distance between solution l and the best solution), $\forall l$.

In a fourth variant (Frequency – F), desirability factors were weighted with respect to the frequency of appearance of each arc in the best solutions encountered so far. Note that in variants (C) and (H), if solution l is the best solution, its weight is set to 100.

The desirability factor of each arc is then assessed on a *desirability scale* ranging from 0 (do not open the arc) to 1 (absolutely open the arc). We define two thresholds $t_c \leq t_o$ on this scale and perform the following comparisons:

- if $0 \leq m_{ij} < t_c$, the arc is closed in the new solution;
- if $t_c \leq m_{ij} \leq t_o$, the arc is undecided;
- if $t_o < m_{ij} \leq 1$, the arc is open in the new solution.

To complete this solution, we first solve a modified CMCF problem using CPLEX. In this modified CMCF, the capacity of closed arcs is set to 0, while the variable cost for undecided arcs is set to $(f_{ij}/u_{ij}) + c_{ij}$, i.e., the cost of arcs in the linear relaxation of CMND. The optimal solution of this problem is then used to create a feasible solution for CMNF by opening all arcs on which there is flow. The cycle-based tabu search is launched from that solution in the hope of finding an improved solution. In a variant of the algorithm, an *intensification* phase is applied to the best solution obtained by TS. This intensification phase is similar to the one described in the original tabu search procedure of Ghamlouche, Crainic, and Gendreau [6]; it involves iteratively modifying the flow distribution of a single commodity at the time and only accepting improving moves. If the best solution found by the above procedure is better than the worst solution in the reference set, it replaces it.

4 Computational results

Computational experiments were performed to first identify good parameter values, and then to evaluate the performance of the method by comparing the results that it produces with those of the path relinking heuristic.

For these experiments, we used one of the original data sets used by Ghamlouche, Crainic, and Gendreau to test the tabu search heuristic [6] and the path relinking method [7]. The 43 problems in this set are general transshipment networks with no parallel arcs. Each commodity corresponds to a single origin-destination pair. On each arc, routing costs are the same for all commodities. Problem instances have been generated to offer for each network size (20 to 100 nodes, 100 to 700 arcs, 10 to 400 commodities), a variety of fixed cost to routing cost ratios and capacity to demand ratios. Each instance is thus uniquely identified by a label consisting of five entries:

1. Number of nodes,
2. Number of arcs,
3. Number of commodities,
4. A letter indicating if variable (V) or fixed (F) costs are dominant in the objective,
5. A letter indicating if capacity constraints are tight (T) or loose (L).

Instance difficulty is largely driven by the number of commodities involved. For a given number of commodities, instance difficulty increases in general with the number of nodes and arcs. For a given network size and number of commodities, problems with tight capacities (T) and where fixed costs dominate (F) are the most difficult. A detailed description of problem instances is given in Crainic, Frangioni, and Gendron [3]. The problem generators as well as the problem instances can be obtained from the authors.

The computer code is written in C++. The exact evaluation of the CMNF problems is done using the LP solver of CPLEX 7.5. All tests were conducted on one 400MHz processor of a 64-processor Sun Enterprise 10000 with 64 Gigabyte of RAM, operating under Solaris 8.

Preliminary testing was performed to find good parameter values on a subset of 10 representative instances. In particular, two values were tested for (t_c, t_o): (0.25, 0.75) and (0.4, 0.6). These tests indicated that the second combination performed significantly better and all further tests were conducted with $(t_c, t_o) = (0.4, 0.6)$. Extensive computational experiments were performed for several values of N using the four variants for combinations, with or without intensification. Different versions of the basic method involving different ways of building the reference set were also considered and tested. The analysis of the results of these various runs, some of which will not be reported in detail here, allowed several conclusions to be drawn:

1. The use of intensification does not significantly improve the results obtained.
2. The best values of N range between 3 and 5; in particular, using $N = 2$ produces markedly inferior results.
3. It is important to use a sufficiently large reference set and to ensure that it is full at initialization step.
4. The combination rule based on frequencies (F) is clearly inferior to the three other ones.

On the basis of these conclusions, we now report on the most interesting variants and combinations. In these, intensification is not used and the size of the reference set is fixed

to 20, a value that provides sufficient diversity in the set. Furthermore, as indicated in section 3, at initialization, if less than 20 local optima have been identified by tabu search, the reference set is filled with the best solutions encountered by tabu search that were not local optima, since preliminary testing showed that this had a positive impact on results. In Table 1, we report the percentage gaps observed between the solutions obtained with scatter search for $N = 3, 4, 5$ and for combination rules (V), (C) and (H) with those produced by path relinking [7] for the 43 instances tested. These gaps are computed as the difference between the value of the scatter search solutions and the average value of the path relinking solutions divided by this path relinking value; negative gaps indicate that scatter search produced a better solution than path relinking. Summary statistics for all 43 runs are provided in Table 2.

These tables show that the performance of the scatter search heuristic varies quite significantly from one instance to another (with gaps ranging from -3.51 to 10.07%) and sometimes between variants for a given instance (see, e.g. instance (100,400,30,F,L) for which gaps range from -2.11 to 5.42%). In general, scatter search does fairly well, but no single variant outperforms path relinking on average. It is interesting to note that, with an average gap of 0.47%, the variant that displays the best overall performance is ($N = 3$, V), which is the "simplest" one, since it requires less solutions than others and combines them in the most straightforward way.

More detailed statistics by problem class (see Table 3) and by problem size (see Table 4) confirm the slight superiority, on average, of the Voting combination rule over the others, but also highlight the fact that there are problem classes and problem sizes where combining more than 3 solutions pays off. Furthermore, one may notice that there are indeed problem classes ((F, L) instances for ($N = 4$, V)) and problem sizes (100–200 commodities for ($N = 3$, V or C)) for which scatter search can do better on average than path relinking.

A more interesting conclusion can be drawn, however, by considering the minimum gap observed over all 9 runs reported here for a given instance: on average, it is equal to -0.37%. By considering multiple runs of scatter search, it is thus possible to obtain better results on average than with path relinking. We pushed this analysis further to determine what could be said about the best solutions obtained by considering the three combinations for a given value of N or the three values of N for any combination rule. In all cases, we observed, unfortunately, positive average gaps. Therefore, if we consider independent runs of scatter search, we probably need to combine all 9 variants to outperform path relinking. There is, however, a good reason for not accepting this as a final answer in the search for methods capable of doing better than path relinking and this is running times. These running times (in CPU seconds) are reported for the case $N = 3$ in Table 5; similar, but in general somewhat longer times were observed for $N = 4$ or 5. As one may easily remark, running times are reasonable for the smaller instance sizes, but they grow quite rapidly. Moreover, one must also note that the running times for scatter search are, except for a very few cases, consistently higher (often 3 to 5 times higher) than the times required by path relinking.

Regarding running times, it is interesting to examine the breakdown of CPU times with

Table 1: Percentage gaps between the scatter search and the path relinking heuristics

Problem	$N = 3$			$N = 4$			$N = 5$		
	(V)	(C)	(H)	(V)	(C)	(H)	(V)	(C)	(H)
25,100,10,V,L	0.76	0.69	0.00	0.00	0.69	0.00	0.69	0.76	0.00
25,100,10,F,L	0.75	0.75	-0.54	0.75	0.75	-0.54	1.65	-1.81	-0.54
25,100,10,F,T	1.20	2.89	-0.33	2.02	2.65	0.36	3.79	0.81	2.76
100,400,10,V,L	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07	0.07
100,400,10,F,L	1.65	1.88	1.83	-1.21	0.07	-1.21	-0.22	-0.16	-0.22
100,400,10,F,T	7.54	9.55	10.07	7.01	8.58	9.21	9.75	6.40	7.22
25,100,30,F,L	-1.81	0.86	-0.09	-0.66	0.24	0.80	0.23	1.69	2.90
25,100,30,V,T	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
25,100,30,F,T	2.62	2.02	2.02	0.79	1.35	1.21	1.29	1.45	1.82
100,400,30,F,L	0.94	1.03	1.52	1.76	-1.60	2.42	-0.84	5.42	-2.11
100,400,30,V,T	-0.03	-0.03	-0.03	-0.03	-0.02	-0.03	-0.02	-0.03	-0.02
100,400,30,F,T	1.04	1.06	1.01	0.51	0.73	0.33	0.33	1.29	0.41
20,230,40,V,L	0.23	0.33	0.23	0.24	0.33	0.43	0.23	0.33	0.23
20,230,40,V,T	0.07	0.07	0.07	0.06	0.07	0.07	0.06	0.07	-0.01
20,230,40,F,T	0.62	0.66	0.80	0.70	0.70	0.70	0.80	0.82	0.82
20,300,40,V,L	0.01	0.19	0.10	0.16	0.10	0.09	0.05	0.17	0.00
20,300,40,F,L	0.27	0.99	0.87	0.74	0.99	0.99	0.99	0.82	0.99
20,300,40,V,T	0.18	0.17	0.17	0.17	0.17	0.18	0.18	0.21	0.17
20,300,40,F,T	0.38	0.47	0.56	0.25	0.64	0.92	0.80	-0.03	0.24
30,520,100,V,L	1.52	2.01	2.70	2.73	2.20	2.62	2.87	3.04	2.72
30,520,100,F,L	-0.11	-0.11	0.13	0.24	-0.35	1.02	0.84	-1.81	0.86
30,520,100,V,T	1.54	1.89	1.64	1.79	1.80	1.81	1.54	1.88	1.66
30,520,100,F,T	4.68	4.92	4.24	1.54	4.68	3.72	3.39	4.98	5.79
30,700,100,V,L	1.22	1.22	1.22	1.22	1.22	1.22	1.42	1.22	1.22
30,700,100,F,L	1.80	1.58	1.89	0.29	3.06	1.86	0.46	1.55	2.36
30,700,100,V,T	1.39	1.39	2.08	1.84	1.87	1.89	1.66	2.05	1.91
30,700,100,F,T	2.04	1.60	1.21	1.81	1.84	2.06	2.55	1.76	2.55
20,230,200,V,L	-2.81	-1.28	-1.40	-1.03	-0.77	0.75	-0.54	0.06	1.00
20,230,200,F,L	-1.84	-3.26	-1.82	-1.72	-2.08	-1.46	-1.81	-0.84	-1.81
20,230,200,V,T	-1.53	-2.09	1.05	-0.26	-0.65	0.75	-0.77	-0.96	1.30
20,230,200,F,T	-0.61	-3.26	-0.97	-1.09	-1.05	-0.54	-0.86	-0.21	-0.53
20,300,200,V,L	-2.80	-1.67	-0.47	-1.26	-1.75	-1.42	-0.21	-0.67	-1.31
20,300,200,F,L	-1.04	-0.69	0.12	-1.44	-1.14	-0.92	1.18	-2.21	2.01
20,300,200,V,T	-2.23	-1.69	1.05	0.42	-1.64	0.42	0.86	1.18	-0.97
20,300,200,F,T	-3.10	-2.90	-1.94	-2.88	-2.50	-3.41	-1.89	-3.51	-2.31
30,520,400,V,L	1.38	0.52	-0.84	0.38	0.26	1.45	0.97	0.57	0.10
30,520,400,F,L	-0.40	-0.92	-0.38	-1.71	-0.80	-1.91	-1.48	-0.73	0.68
30,520,400,V,T	0.92	0.84	0.93	1.02	0.89	1.21	1.30	1.04	0.81
30,520,400,F,T	-0.32	-0.28	1.16	0.88	0.88	0.82	-0.20	-0.20	0.54
30,700,400,V,L	1.55	1.40	4.51	3.15	1.70	2.90	2.21	2.34	3.97
30,700,400,F,L	0.62	-1.02	3.39	-0.88	1.72	0.62	-1.06	0.60	0.21
30,700,400,V,T	1.78	1.47	-0.08	1.59	1.64	1.18	1.46	1.44	1.37
30,700,400,F,T	0.27	0.08	0.94	1.60	0.84	1.68	-0.20	1.76	1.04

Table 2: Percentage gaps between the scatter search and the path relinking heuristics – Global statistics

	$N = 3$			$N = 4$			$N = 5$		
	(V)	(C)	(H)	(V)	(C)	(H)	(V)	(C)	(H)
Average	0.47	0.54	0.90	0.50	0.66	0.80	0.78	0.76	0.93
Std. dev.	1.88	2.12	1.99	1.61	1.88	1.87	1.89	1.85	1.83
Minimum	-3.10	-3.26	-1.94	-2.88	-2.50	-3.41	-1.89	-3.51	-2.31
Maximum	7.54	9.55	10.07	7.01	8.58	9.21	9.75	6.40	7.22

Table 3: Average percentage gaps by problem class between the scatter search and the path relinking heuristics

Problem		$N = 3$			$N = 4$			$N = 5$		
class	#	(V)	(C)	(H)	(V)	(C)	(H)	(V)	(C)	(H)
V,L	10	0.11	0.35	0.61	0.57	0.41	0.81	0.78	0.79	0.80
F,L	11	0.08	0.10	0.63	-0.35	0.08	0.15	-0.01	0.23	0.48
V,T	10	0.21	0.20	0.69	0.66	0.41	0.75	0.63	0.69	0.62
F,T	12	1.36	1.40	1.56	1.10	1.61	1.42	1.63	1.28	1.70
All	43	0.47	0.54	0.90	0.50	0.66	0.80	0.78	0.76	0.93

Table 4: Average percentage gaps by problem size between the scatter search and the path relinking heuristics

Number of		$N = 3$			$N = 4$			$N = 5$		
commodities	#	(V)	(C)	(H)	(V)	(C)	(H)	(V)	(C)	(H)
10 – 40	19	0.87	1.24	0.96	0.70	0.87	0.84	1.04	0.96	0.78
100 – 200	16	-0.12	-0.15	0.67	0.14	0.30	0.65	0.67	0.47	1.03
400	8	0.73	0.26	1.20	0.75	0.89	0.99	0.38	0.85	1.09
All	43	0.47	0.54	0.90	0.50	0.66	0.80	0.78	0.76	0.93

Table 5: Running times (in seconds) for the scatter search ($N = 3$) and the path relinking heuristics

Problem	Scatter search			Path Relinking
	(V)	(C)	(H)	
25,100,10,V,L	30	16	50	13
25,100,10,F,L	84	91	79	16
25,100,10,F,T	107	43	152	25
100,400,10,V,L	347	1044	419	99
100,400,10,F,L	796	641	710	112
100,400,10,F,T	2,475	2,351	1,413	201
25,100,30,F,L	431	282	349	79
25,100,30,V,T	69	42	124	93
25,100,30,F,T	90	58	211	100
100,400,30,F,L	2,959	4,279	2,137	301
100,400,30,V,T	927	2,279	3,500	451
100,400,30,F,T	3,886	2,623	2,581	579
20,230,40,V,L	258	148	507	132
20,230,40,V,T	385	418	514	149
20,230,40,F,T	198	440	305	146
20,300,40,V,L	447	784	719	247
20,300,40,F,L	464	117	913	241
20,300,40,V,T	218	620	965	246
20,300,40,F,T	534	733	751	138
30,520,100,V,L	13,840	3,057	13,187	1,351
30,520,100,F,L	9,932	10,400	19,370	1,843
30,520,100,V,T	3,817	6,090	5,902	1,423
30,520,100,F,T	10,256	13,336	12,196	1,371
30,700,100,V,L	6,453	6,860	14,220	1,899
30,700,100,F,L	8,707	18,718	16,131	2,190
30,700,100,V,T	7,489	7,933	13,755	1,674
30,700,100,F,T	4,948	5,244	7,211	1,765
20,230,200,V,L	7,943	7,413	10,119	2,035
20,230,200,F,L	3,080	7,011	6,090	2,508
20,230,200,V,T	4,412	6,116	5,814	1,946
20,230,200,F,T	6,390	10,082	7,919	2,954
20,300,200,V,L	11,017	11,583	11,238	3,561
20,300,200,F,L	8,276	10,416	8,026	3,913
20,300,200,V,T	11,088	12,352	18,681	3,860
20,300,200,F,T	7,365	9,832	9,318	4,001
30,520,400,V,L	29,581	35,185	101,605	31,546
30,520,400,F,L	104,151	90,483	113,216	35,671
30,520,400,V,T	15,674	38,438	39,859	23,546
30,520,400,F,T	67,333	94,994	83,548	60,123
30,700,400,V,L	100,944	112,100	74,263	19,433
30,700,400,F,L	107,977	131,285	120,494	58,762
30,700,400,V,T	31,122	88,147	115,992	32,450
30,700,400,F,T	111,778	93,911	106,186	51,235

Table 6: Number of iterations, new solutions, and reference set updates ($N = 3$)

Problem	(V)			(C)			(H)		
	#iter	new	upd.	#iter	new	upd.	#iter	new	upd.
100,400,10,F,L	68	57	11	49	42	8	47	41	9
100,400,10,F,T	72	64	13	51	48	10	35	34	6
100,400,10,V,L	22	19	3	56	45	16	23	23	4
100,400,30,F,L	66	56	13	102	92	24	43	37	12
100,400,30,F,T	51	50	10	32	29	7	33	30	7
100,400,30,V,T	12	12	2	27	26	7	39	34	9
25,100,10,F,L	73	60	13	73	60	13	61	51	11
25,100,10,F,T	58	50	12	20	19	4	78	71	15
25,100,10,V,L	35	25	6	16	13	3	47	39	9
25,100,30,F,L	150	143	32	85	83	18	105	103	20
25,100,30,F,T	24	23	5	15	14	2	49	47	16
25,100,30,V,T	19	15	3	10	10	3	29	28	8
20,230,40,V,L	20	20	6	11	11	2	35	35	7
20,230,40,V,T	28	28	7	28	28	7	33	33	8
20,230,40,F,T	15	15	4	29	28	6	19	19	5
20,230,200,V,L	116	113	25	95	92	31	115	114	28
20,230,200,F,L	43	43	11	96	92	20	71	70	15
20,230,200,V,T	66	66	13	85	81	20	70	68	14
20,230,200,F,T	90	89	20	126	121	35	93	87	17
20,300,40,V,L	23	22	5	35	32	11	32	30	6
20,300,40,F,L	22	20	4	6	6	0	38	35	8
20,300,40,V,T	11	11	2	25	25	7	36	35	9
20,300,40,F,T	25	23	5	32	30	7	31	30	6
20,300,200,V,L	114	113	26	115	109	26	94	93	19
20,300,200,F,L	85	84	18	100	94	26	66	63	15
20,300,200,V,T	114	111	24	113	107	29	166	163	36
20,300,200,F,T	83	81	15	97	96	21	82	81	16
30,520,100,V,L	94	84	26	21	20	5	82	74	18
30,520,100,F,L	73	72	14	73	72	14	134	134	23
30,520,100,V,T	29	28	8	44	41	11	39	38	10
30,520,100,F,T	81	79	22	99	93	26	81	79	18
30,520,400,V,L	49	49	15	49	42	15	124	117	30
30,520,400,F,L	183	183	38	130	128	26	137	136	27
30,520,400,V,T	27	25	6	54	47	12	48	47	11
30,520,400,F,T	106	106	24	125	115	27	88	87	19
30,700,100,V,L	30	29	9	31	30	7	57	51	14
30,700,100,F,L	42	37	11	85	77	21	70	68	12
30,700,100,V,T	32	32	8	32	32	8	53	53	10
30,700,100,F,T	24	21	5	25	21	5	31	27	6
30,700,400,V,L	119	118	25	134	128	29	61	58	14
30,700,400,F,L	125	124	29	142	138	30	109	108	20
30,700,400,V,T	35	35	10	96	90	21	120	116	23
30,700,400,F,T	148	142	36	92	90	17	89	87	19

respect to the different steps of the proposed algorithm. Detailed statistics were collected for a representative subset of 10 instances using profiling software. These statistics clearly show that the tabu search step is the most time-consuming one: on average, it requires 80% of the CPU time. This figure somewhat varies among instances and it can be observed that the instances that are solved faster than comparable-size ones (e.g., (30,520,400,V,T)) have a smaller fraction of the CPU time used devoted to tabu search. This simply reflects the fact that these instances are require less CPU time, **because** the tabu search step is, in general, shorter. The initialization phase of the algorithm (i.e., steps 1 and 2) takes on average 5% of the total running time and the remainder of the procedure (steps 3, 4, 5, and 7) the last 15%.

Another critical issue with respect to the performance of a scatter search procedure is its ability to explore new, meaningful portions of the solution space, since it is in these portions that one hopes to find better solutions. Table 6 provides detailed statistics on the search performed when $N = 3$ for combination rules (V), (C) and (H) on all 43 instances of the benchmark. For each instance, we report the number of iterations performed, the number of new solutions (i.e., solutions not present in the reference set) generated, and the number of times the reference set was updated (because the current solution was better than the worst solution in the reference set). On average, more than 60 iterations of the procedure are performed and more than 90% of these iterations yield "new" solutions, while the reference set is updated in more than 20% of iterations. This confirms that the procedure is consistently able to yield new solutions of high quality.

5 Conclusion and future research directions

In this paper, we have proposed a scatter search heuristic for the fixed-charge capacitated multicommodity network design problem (CMND). As far as we know, this is the first time that scatter search is applied to the generic CMND formulation. Our heuristic, which is based upon the integer design variables, allows for several variants that use different rules for combining previously obtained solutions. Extensive computational experiments performed on a fairly large set of benchmark problems have shown that, on average, the most effective variants of the scatter search heuristic do not perform better than the best existing method for the problem (a path relinking approach), but that they do come very close.

Further analysis of the computational results has highlighted the fact that multiple runs of scatter search could lead to better results than path relinking, but at the price of very high computational requirements. We believe, however, that a more refined parallel scatter search approach, involving for instance parallel search threads which exchange meaningful information, could better exploit the full potential of scatter search and thus prove much more effective (see, e.g., [1]). We intend to start investigating such an approach in the very near future.

Acknowledgments

Funding for this project has been provided by the Natural Sciences and Engineering Council of Canada through its Research Grant program. This support is gratefully acknowledged by the authors. We also want to thank Ms. Geneviève Hernu for the care and attention she displayed in the programming and testing of the scatter search algorithm.

References

[1] Crainic, T. G. and M. Toulouse (2003), "Parallel Strategies for Meta-heuristics", in F. Glover and G. Kochenberger (eds.), *Handbook of Metaheuristics*, Kluwer Academic Publishers, Boston, USA, 475–513.

[2] Balakrishnan, A., T.L. Magnanti, and P. Mirchandani (1997), "Network Design", in M. Dell'Amico, F. Maffioli, and S. Martello (eds.), *Annotated Bibliographies in Combinatorial Optimization*, Wiley, New York, USA, 311–334.

[3] Crainic, T.G., A. Frangioni, and B. Gendron (2001), "Bundle-Based Relaxation Methods for Multicommodity Capacitated Network Design", *Discrete Applied Mathematics* **112**, 73–99.

[4] Crainic, T.G., M. Gendreau, and J.M. Farvolden (2000), "A Simplex-Based Tabu Search Method for Capacitated Network Design", *INFORMS Journal on Computing* **12**, 223–236.

[5] Gendron, B., T.G. Crainic, and A. Frangioni (1998), "Multicommodity Capacitated Network Design", in B. Sansó and P. Soriano (eds.), *Telecommunications Network Planning*, Kluwer Academic Publishers, Boston, USA, 1–19.

[6] Ghamlouche, I., T.G. Crainic, and M. Gendreau (2003), "Cycle-based Neighbourhoods for Fixed-Charge Capacitated Multicommodity Network Design", *Operations Research* **51**, 655–667.

[7] Ghamlouche, I., T.G. Crainic, and M. Gendreau (2004), "Path Relinking, Cycle-based Neighbourhoods and Capacitated Multicommodity Network Design", *Annals of Operations Research* **131**, 109–133.

[8] Glover, F. (1977), "Heuristics for Integer Programming using Surrogate Constraints", *Decision Sciences* **8**, 156–166.

[9] Glover, F. (1998), "A Template for Scatter Search and Path Relinking", in J. Hao, E. Lutton, E. Ronald, M. Schoenauer, and D. Snyers (eds.), *Artificial Evolution*, (Lecture Notes in Computer Science, Vol. 1363), Springer, Berlin, Germany, 13–54.

[10] Glover, F., and M. Laguna (1997), *Tabu Search*, Kluwer Academic Publishers, Boston, USA.

[11] Glover, F., M. Laguna, and R. Martí (2000), "Fundamentals of Scatter Search and Path Relinking", *Control and Cybernetics* **39** (3), 653–684.

[12] Holmberg, K. and D. Yuan (2000), "A Lagrangean Heuristic Based Branch-and-Bound Approach for the Capacitated Network Design Problem", *Operations Research* **48** (3), 461–481.

[13] Laguna, M. and R. Martí (2003), *Scatter Search: Methodology and Implementations in C*, Kluwer Academic Publishers, Boston, USA.

[14] Magnanti, T.L. and R.T. Wong (1986), "Network Design and Transportation Planning: Models and Algorithms", *Transportation Science* **18** (1), 1–55.

[15] Minoux, M. (1986), "Network Synthesis and Optimum Network Design Problems: Models, Solution Methods and Applications", *Networks* **19**, 313–360.